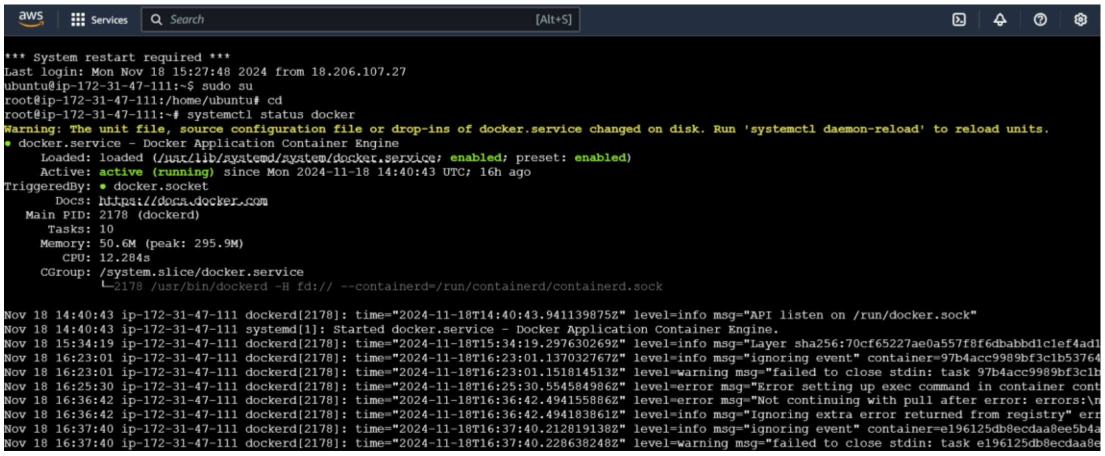
Docker Network and Compose

To create and configure a Docker network on an EC2 instance, follow these steps:

1. Install Docker on the EC2 Instance

If Docker is not already installed on your EC2 instance, install it.
Start and enable the Docker service.

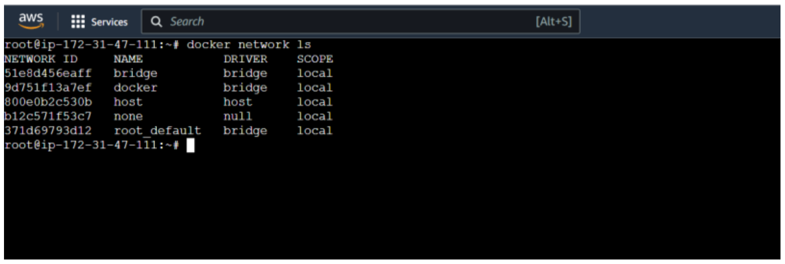


```
*** System restart required ***
Last login: Mon Nov 18 15:27:48 2024 from 18.206.107.27
ubuntu@ip-172-31-47-111:~$ sudo su
root@ip-172-31-47-111:/home/ubuntu# cd
root@ip-172-31-47-111:~# systemctl status docker
Warning: The unit file, source configuration file or drop-ins of docker.service changed on disk. Run 'systemctl daemon-reload' to reload units.
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-11-18 14:40:43 UTC; 16h ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2178 (dockerd)
      Tasks: 10
     Memory: 50.6M (peak: 295.9M)
        CPU: 12.284s
     CGroup: /system.slice/docker.service
             └─2178 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Nov 18 14:40:43 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T14:40:43.941139875Z" level=info msg="API listen on /run/docker.sock"
Nov 18 14:40:43 ip-172-31-47-111 systemd[1]: Started docker.service - Docker Application Container Engine.
Nov 18 15:34:19 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T15:34:19.297630269Z" level=info msg="Layer sha256:70cf65227ae0a557f8f6dbabbd1c1ef4ad1
Nov 18 16:23:01 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T16:23:01.137032767Z" level=info msg="ignoring event" container=97b4acc9989bf3c1b53764
Nov 18 16:23:01 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T16:23:01.151814513Z" level=warning msg="failed to close stdin: task 97b4acc9989bf3c1b
Nov 18 16:25:30 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T16:25:30.554584986Z" level=error msg="Error setting up exec command in container cont
Nov 18 16:36:42 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T16:36:42.494155886Z" level=error msg="Not continuing with pull after error: errors:\n
Nov 18 16:36:42 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T16:36:42.494183861Z" level=info msg="Ignoring extra error returned from registry" err
Nov 18 16:37:40 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T16:37:40.212819138Z" level=info msg="ignoring event" container=e196125db8ecdaa8ee5b4a
Nov 18 16:37:40 ip-172-31-47-111 dockerd[2178]: time="2024-11-18T16:37:40.228638248Z" level=warning msg="failed to close stdin: task e196125db8ecdaa8e
```

2. Create a Docker Network

Docker networks allow containers to communicate with each other.
Use the following command to create a custom bridge network:
Replace docker with the desired name of your network.
This creates a bridge network by default, which is ideal for most use cases.



```
root@ip-172-31-47-111:~# docker network ls
NETWORK ID     NAME           DRIVER    SCOPE
51e8d456eaff   bridge         bridge    local
9d751f13a7ef   docker         bridge    local
800e0b2c530b   host           host      local
b12c571f53c7   none           null      local
371d69793d12   root_default   bridge    local
root@ip-172-31-47-111:~#
```

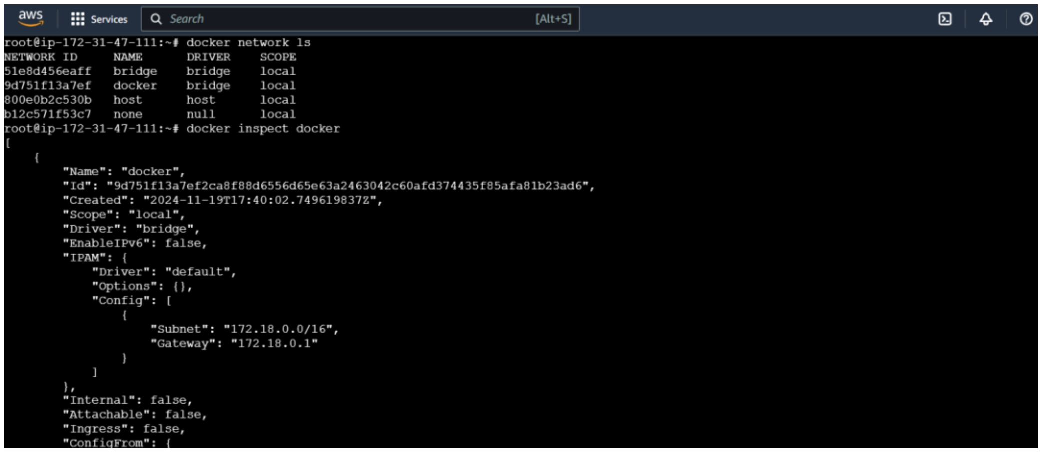
3. Verify the Network

To confirm the network was created, run: - sudo docker network ls

4. Run Containers in the Network

To connect containers to this network, use the `--network` flag when running `docker run`.

`Containers in the same network can communicate using their container names.`



```
root@ip-172-31-47-111:~# docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
51e8d456eaff   bridge    bridge    local
9d751f13a7ef   docker    bridge    local
800e0b2c530b   host      host      local
b12c571f53c7   none      null      local
root@ip-172-31-47-111:~# docker inspect docker
[
    {
        "Name": "docker",
        "Id": "9d751f13a7ef2ca8f88d6556d65e63a2463042c60afd374435f85afa81b23ad6",
        "Created": "2024-11-19T17:40:02.749619837Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
```

5. Using Docker Compose with the Network

A `docker-compose.yaml` file is used to define and manage multi-container Docker applications. Here's a basic template for a `docker-compose.yaml`

Key Sections Explained:

`version:`

`Specifies the Docker Compose version. 3.8 is widely compatible with most setups.`

`services:`

`Defines individual containers.`
`Each service represents one container with specific configurations.`

`image:`

Specifies the Docker image to use.

ports:

Maps host ports to container ports.

volumes:

Mounts local directories or creates persistent data storage.

depends_on:

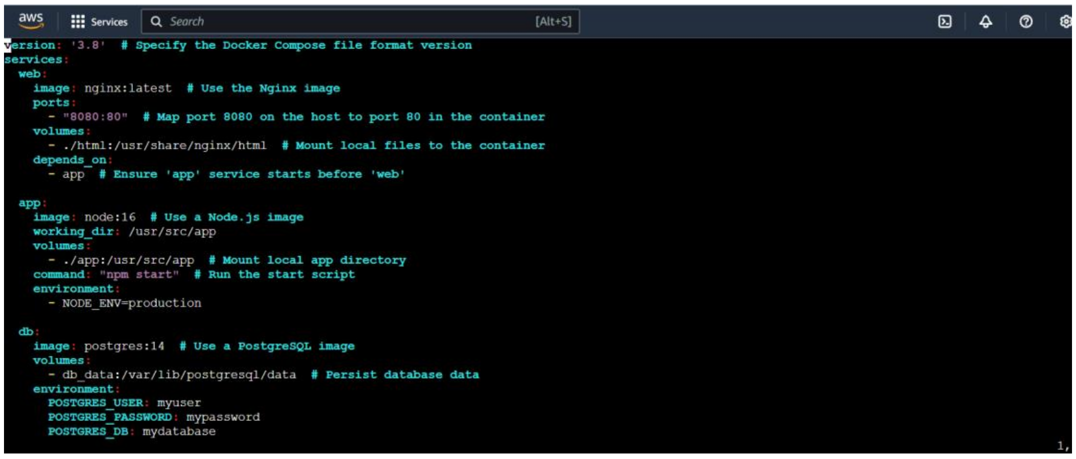
Ensures services start in a defined order.

environment:

Passes environment variables to the container.

volumes: (global)

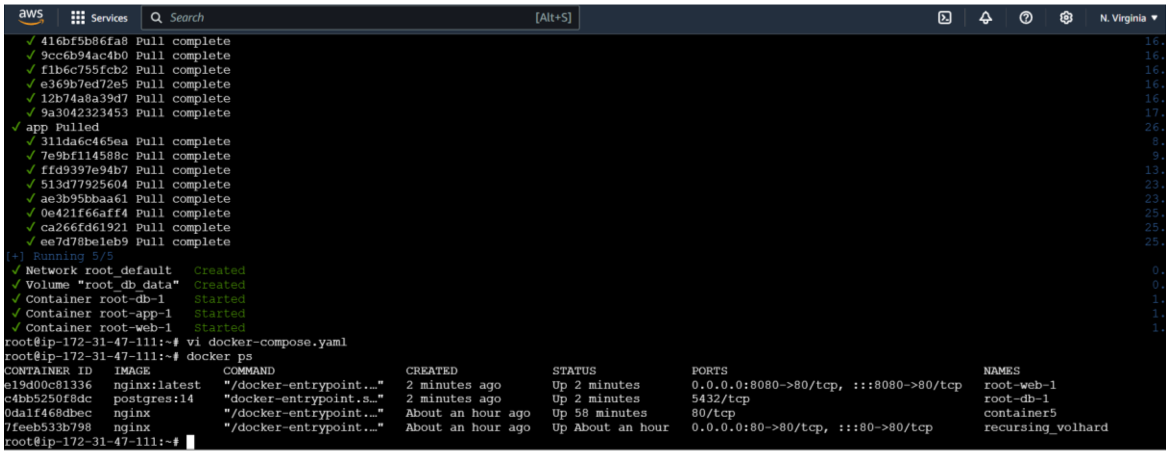
Defines named volumes for persistent storage.



```
version: '3.8'  # Specify the Docker Compose file format version
services:
  web:
    image: nginx:latest  # Use the Nginx image
    ports:
      - "8080:80"  # Map port 8080 on the host to port 80 in the container
    volumes:
      - ./html:/usr/share/nginx/html  # Mount local files to the container
    depends_on:
      - app  # Ensure 'app' service starts before 'web'

  app:
    image: node:16  # Use a Node.js image
    working_dir: /usr/src/app
    volumes:
      - ./app:/usr/src/app  # Mount local app directory
    command: "npm start"  # Run the start script
    environment:
      - NODE_ENV=production

  db:
    image: postgres:14  # Use a PostgreSQL image
    volumes:
      - db_data:/var/lib/postgresql/data  # Persist database data
    environment:
      POSTGRES_USER: myuser
      POSTGRES_PASSWORD: mypassword
      POSTGRES_DB: mydatabase
```



```
   ✓ 416bf5b86fa8 Pull complete
   ✓ 9cc6b94ac4b0 Pull complete
   ✓ f1b6c755fcb2 Pull complete
   ✓ e369b7ed72e5 Pull complete
   ✓ 12b74a8a39d7 Pull complete
   ✓ 9a3042323453 Pull complete
 ✓ app Pulled
   ✓ 311da6c465ea Pull complete
   ✓ 7e9bf114588c Pull complete
   ✓ ffd9397e94b7 Pull complete
   ✓ 513d77925604 Pull complete
   ✓ ae3b95bbaa61 Pull complete
   ✓ 0e421f66aff4 Pull complete
   ✓ ca266fd61921 Pull complete
   ✓ ee7d78be1eb9 Pull complete
[+] Running 5/5
 ✓ Network root_default    Created
 ✓ Volume "root_db_data"   Created
 ✓ Container root-db-1     Started
 ✓ Container root-app-1    Started
 ✓ Container root-web-1    Started
root@ip-172-31-47-111:~# vi docker-compose.yaml
root@ip-172-31-47-111:~# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED             STATUS              PORTS                                   NAMES
e19d00c81336   nginx:latest   "/docker-entrypoint.…"   2 minutes ago       Up 2 minutes        0.0.0.0:8080->80/tcp, :::8080->80/tcp   root-web-1
c4bb5250f8dc   postgres:14    "docker-entrypoint.s…"   2 minutes ago       Up 2 minutes        5432/tcp                                root-db-1
0da1f468dbec   nginx          "/docker-entrypoint.…"   About an hour ago   Up 58 minutes       80/tcp                                  container5
7feeb533b798   nginx          "/docker-entrypoint.…"   About an hour ago   Up About an hour    0.0.0.0:80->80/tcp, :::80->80/tcp       recursing_volhard
root@ip-172-31-47-111:~# 
```

6. Advanced: Configure Network Subnet

If you need to specify a subnet, you can do so during network creation

7. Clean Up

To remove the network when no longer needed.